Advanced Accessibility

Best Practices and new APIs in Snow Leopard

Session 127

These are confidential sessions—please refrain from streaming, blogging, or taking pictures.

At the convention center, a movie will be placed here

Advanced Accessibility

Best Practices and new APIs in Snow Leopard

James Dempsey
Cocoa Frameworks Engineer

Prerequisites

Familiarity with

- Application accessibility on Mac OS X
- Accessibility hierarchy
- Accessibility Inspector
- Voiceover

Enabling Your Mac Applications for Assistive Technologies

North Beach
Wednesday 5:00PM

Introduction

- What's new in Snow Leopard
- Brief technical review
- Making custom views accessible
  - Graphics view in Sketch sample project
- Focus on
  - Design issues
  - Recommended process
  - Troubleshooting

What's New in Snow Leopard

New roles and subroles

- Cell-based tables
  - Addition of new AXCell role, with support for column and row spans
  - New cell-related notifications and attributes for AXTable
- Layout areas
  - Roles for views that do layout or a graphics canvas
- Level indicators
- Content lists

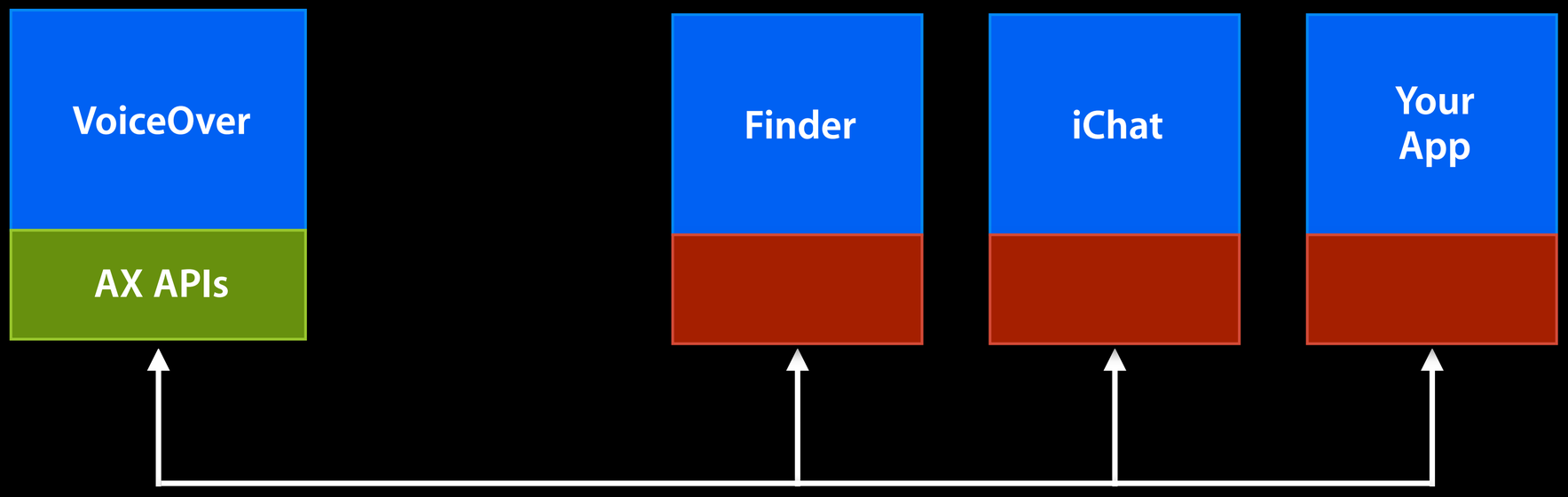
What's New in Snow Leopard

New functionality

- NSImage accessibility descriptions
- NSMenu and NSMenuItem support overridden string attributes
- Optional array attribute methods in NSAccessibility protocol

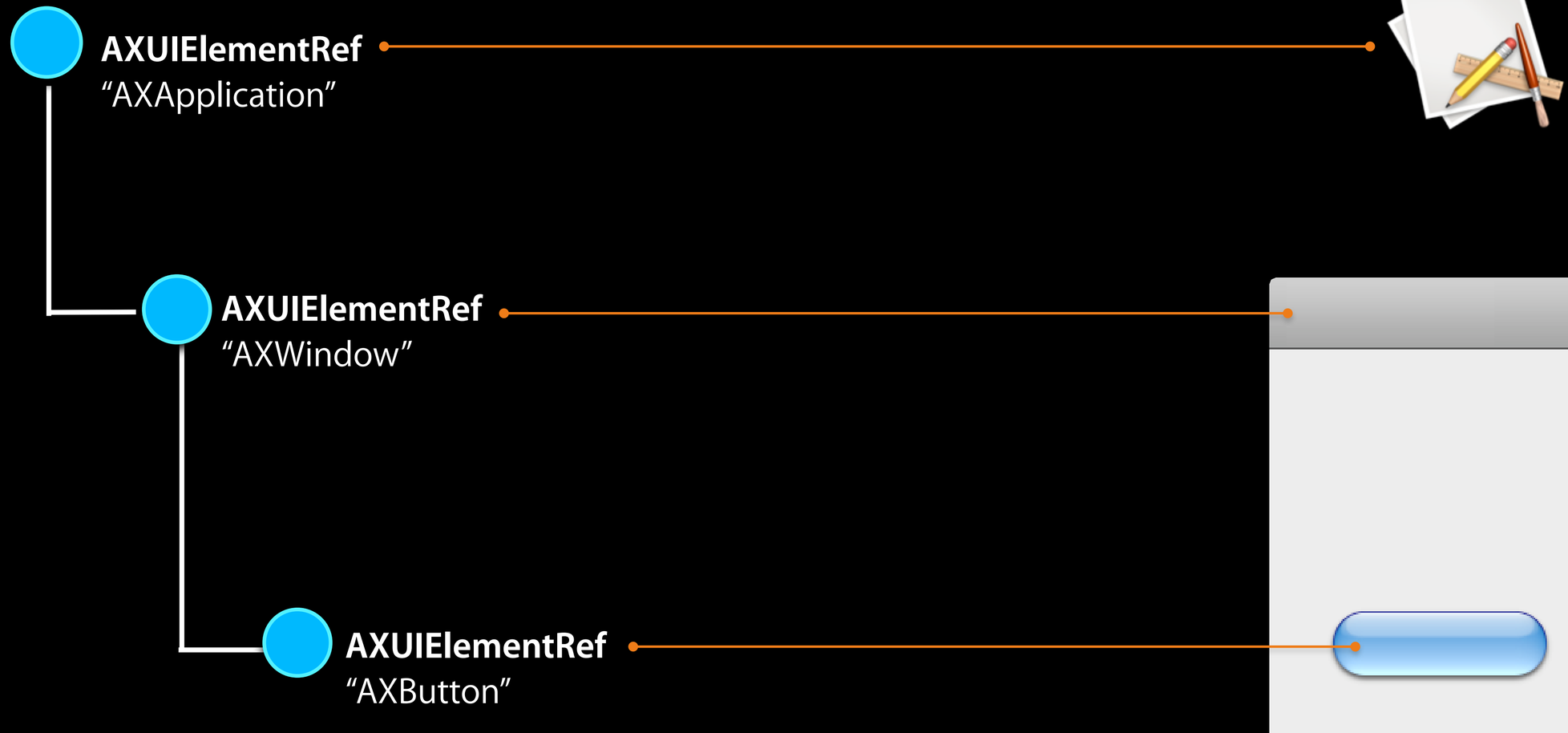
```
- (NSUInteger)accessibilityIndexOfChild:(id)child;

- (NSUInteger)accessibilityArrayAttributeCount:(NSString *)attribute;

- (NSArray *)accessibilityArrayAttributeValues:(NSString *)attribute
                                         index:(NSUInteger)index maxCount:(NSUInteger)maxCount;
```

- Many more noted in AppKit Release Notes


VoiceOver
AX APIs
Finder
iChat
Your App

Apps Vend a Hierarchy of UI Elements

Each node in tree represents a UI element



Apps Vend a Hierarchy of UI Elements

A node can support attributes, actions, and notifications

Each Node Represents an Object

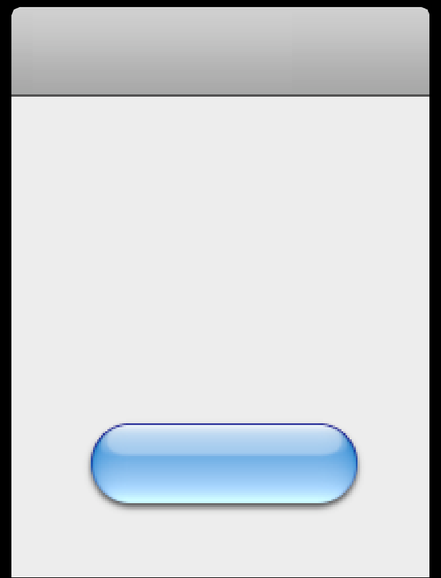
Each object implements the NSAccessibility protocol

AXUIElementRef
"AXApplication"

NSApplication
(NSAccessibility)

AXUIElementRef
"AXWindow"

NSWindow
(NSAccessibility)



AXUIElementRef
"AXButton"

NSButtonCell
(NSAccessibility)

NSAccessibility Protocol

Getting and setting attribute values

```
- (NSArray *)accessibilityAttributeNames;

- (id)accessibilityAttributeValue:(NSString *)attribute;

- (BOOL)accessibilityIsAttributeSettable:(NSString *)attribute;

- (void)accessibilitySetValue:(id)value
                 forAttribute:(NSString *)attribute;
```

NSAccessibility Protocol

Getting parameterized attribute values

- Used in complex UI elements like text, cell-based tables, layout areas

```
- (NSArray *)accessibilityParameterizedAttributeNames;


- (void)accessibilityAttributeValue:(id)attribute
                       forParameter:(id)parameter;
```

NSAccessibility Protocol

Performing actions

```
- (NSArray *)accessibilityActionNames;

- (NSString *)accessibilityActionDescription:(NSString *)action;

- (void) accessibilityPerformAction:(NSString *)action;
```

Accessibility Notifications

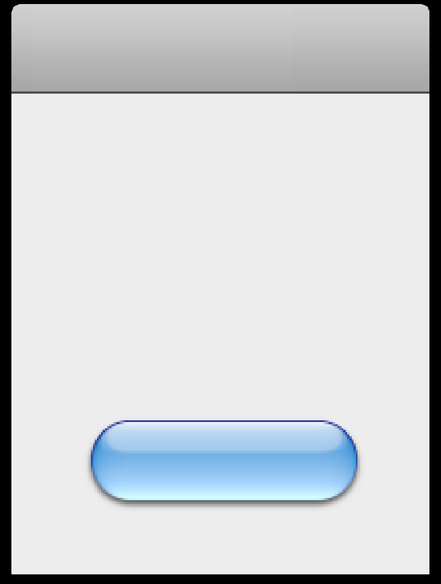
Broadcast important changes in state

- Most handled by framework automatically

```
NSAccessibilityPostNotification(id element, NSString *notification)
```

Clients Can Access Root of Tree

Top-down navigation via AXChildren attribute



Clients Can Access Root of Tree

Top-down navigation via AXChildren attribute

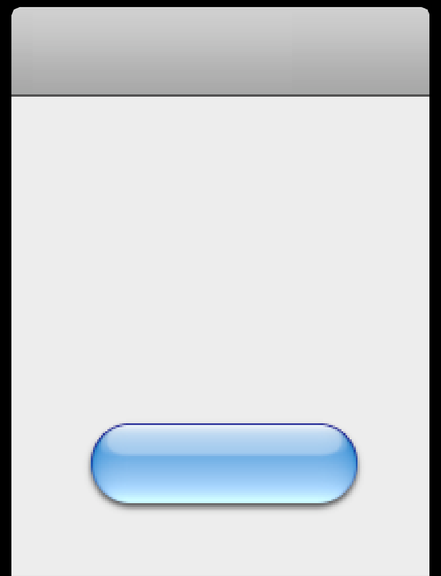
AXUIElementRef
"AXApplication"

NSApplication

AXChildren

AXUIElementRef
"AXWindow"

NSWindow



Clients Can Access Root of Tree

Top-down navigation via AXChildren attribute

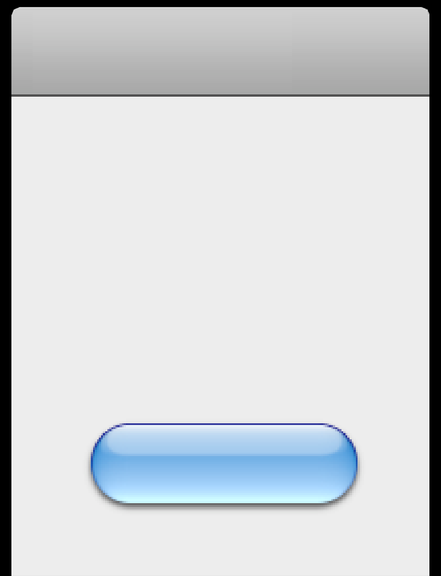
AXUIElementRef
"AXApplication"

NSApplication

AXChildren

AXUIElementRef
"AXWindow"

NSWindow



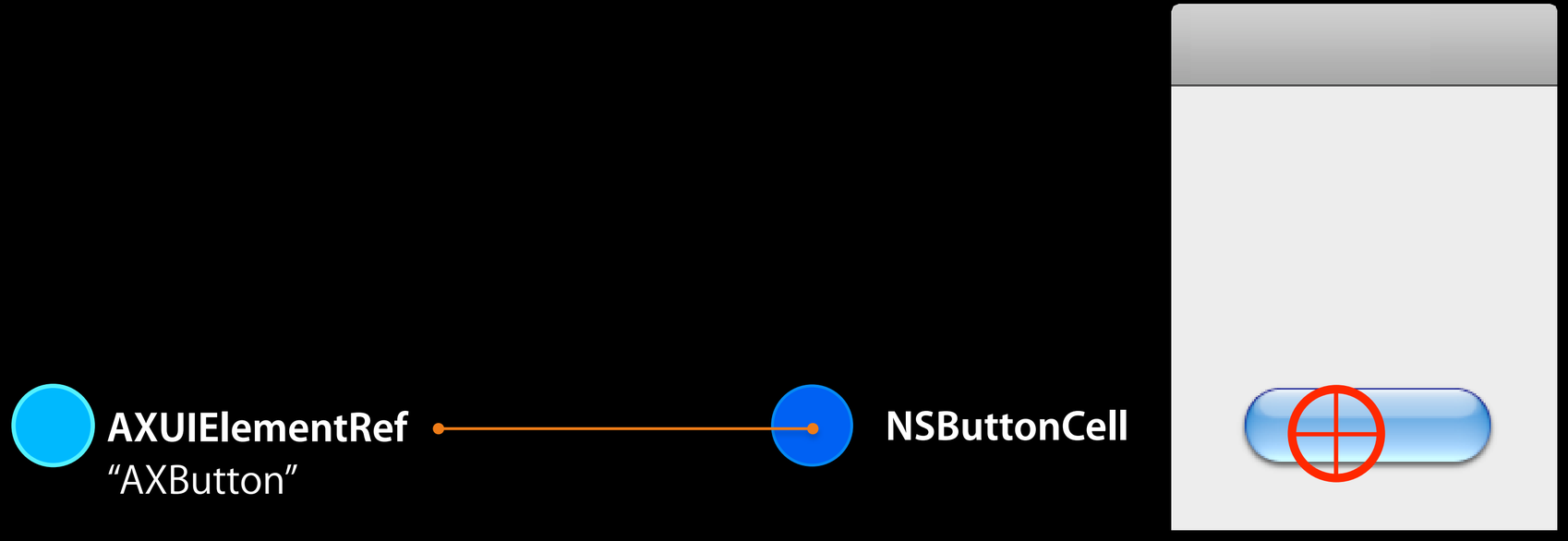
AXChildren

AXUIElementRef
"AXButton"

NSButtonCell

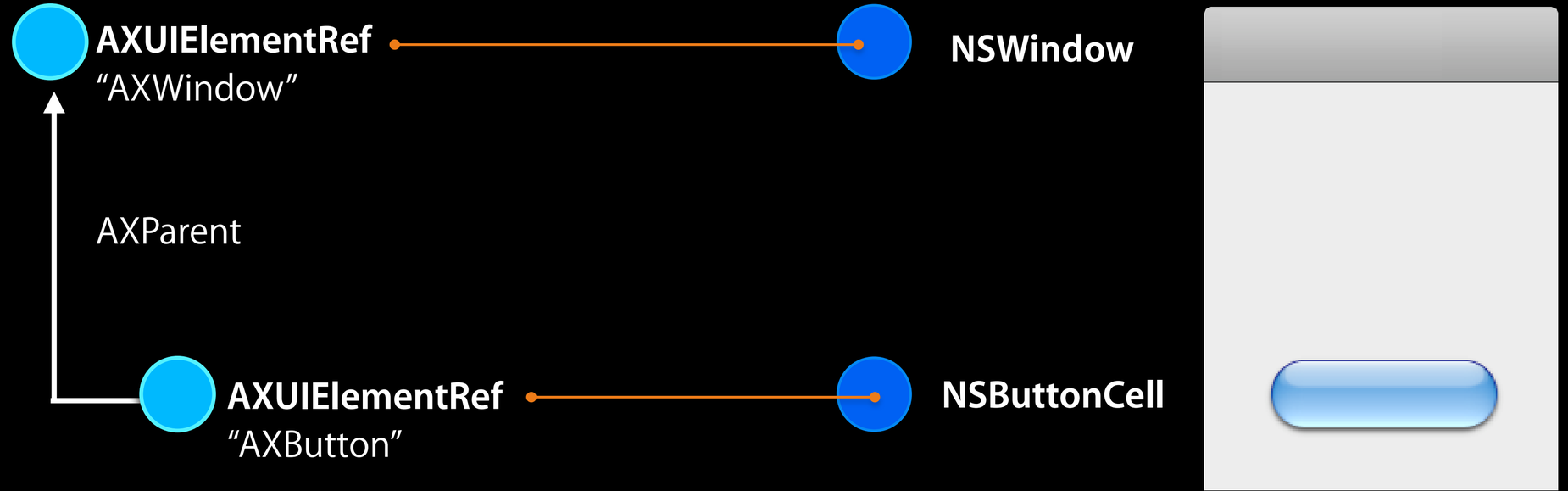
Clients Can Access by Hit Testing

Bottom-up navigation via AXParent attribute



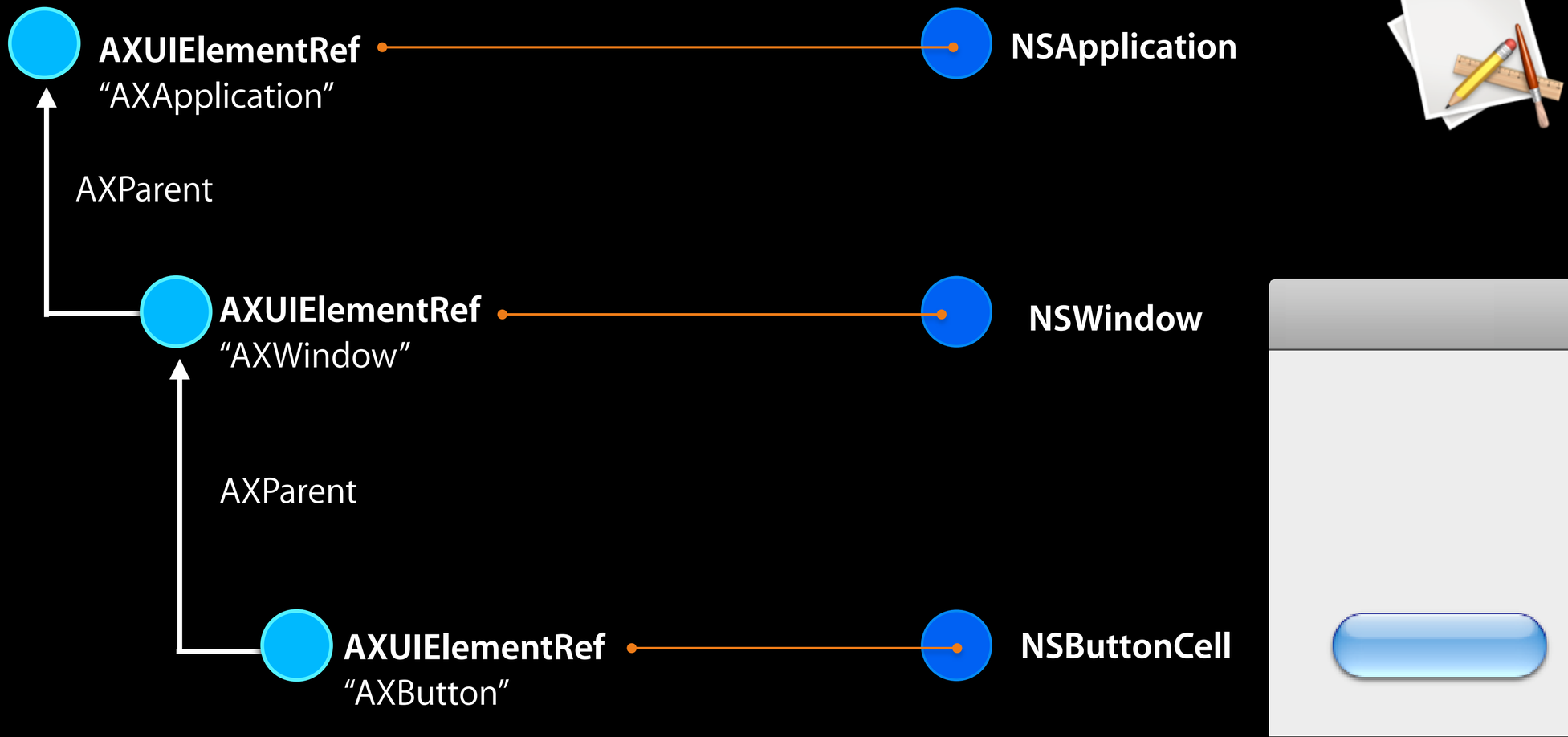
Clients Can Access by Hit Testing

Bottom-up navigation via AXParent attribute



Clients Can Access by Hit Testing

Bottom-up navigation via AXParent attribute



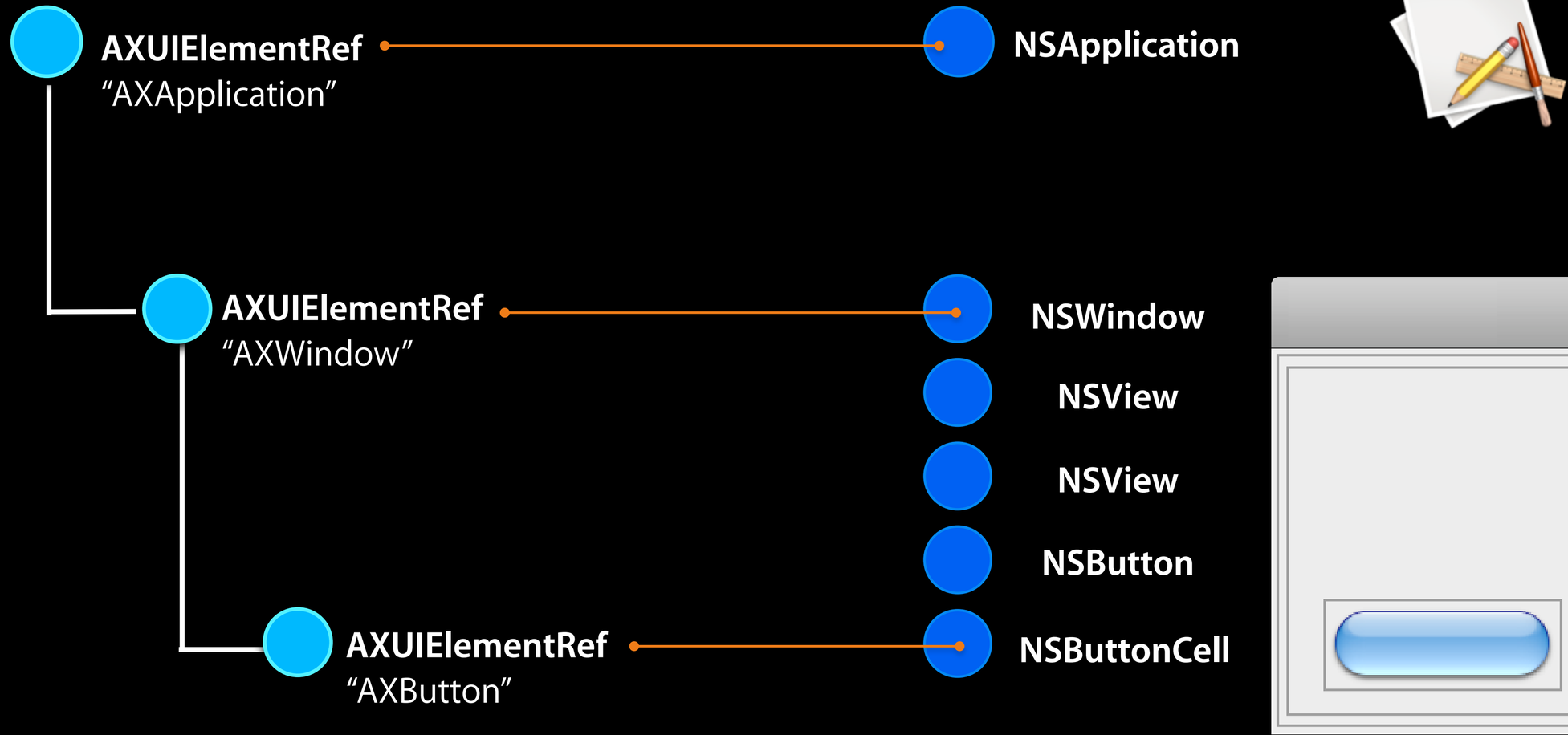
NSAccessibility Protocol

Hit testing, focus testing, is ignored testing

```
- (id)accessibilityHitTest:(NSPoint)point;

- (id)accessibilityFocusedUIElement;

- (BOOL)accessibilityIsIgnored;
```

Not Every View Represented

Simplify the view hierarchy



Some Elements Are Ignored

Ignored is not the same as hidden

AXUIElementRef
"AXApplication"

NSApplication

AXUIElementRef
"AXWindow"

NSWindow

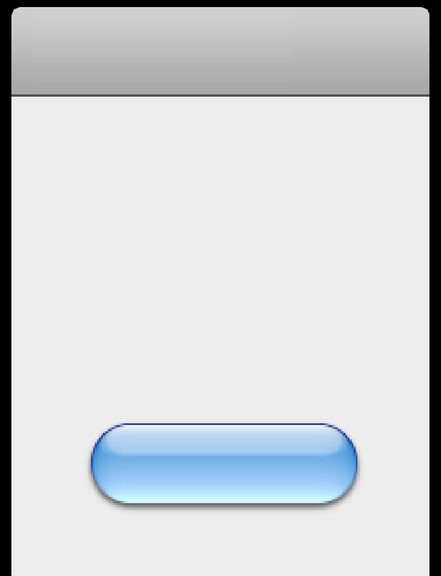
NSView

NSView

NSButton

AXUIElementRef
"AXButton"

NSButtonCell



Views Are Hidden by Default

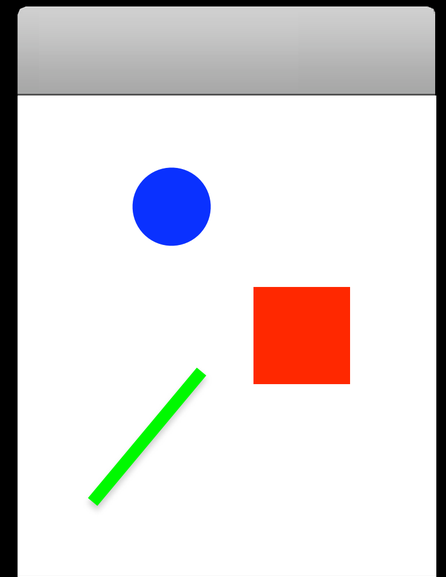
Subclasses of NSControl are hidden by default also

AXUIElementRef
"AXApplication"

NSApplication

AXUIElementRef
"AXWindow"

NSWindow



AXUIElementRef
"AXScrollArea"

NSScrollView

SKTGraphicsView

Demo

Making Custom Views Accessible

- Phase 1: Design the basic structure
  - Determine roles and subroles of elements
  - Map object hierarchy to accessibility hierarchy
- Phase 2: Build the basic structure
  - Parent/child relationships
  - Element size and position
  - Hit testing
- Phase 3: Fill in the remaining pieces
  - Remaining attributes, actions, focus testing
  - Notifications

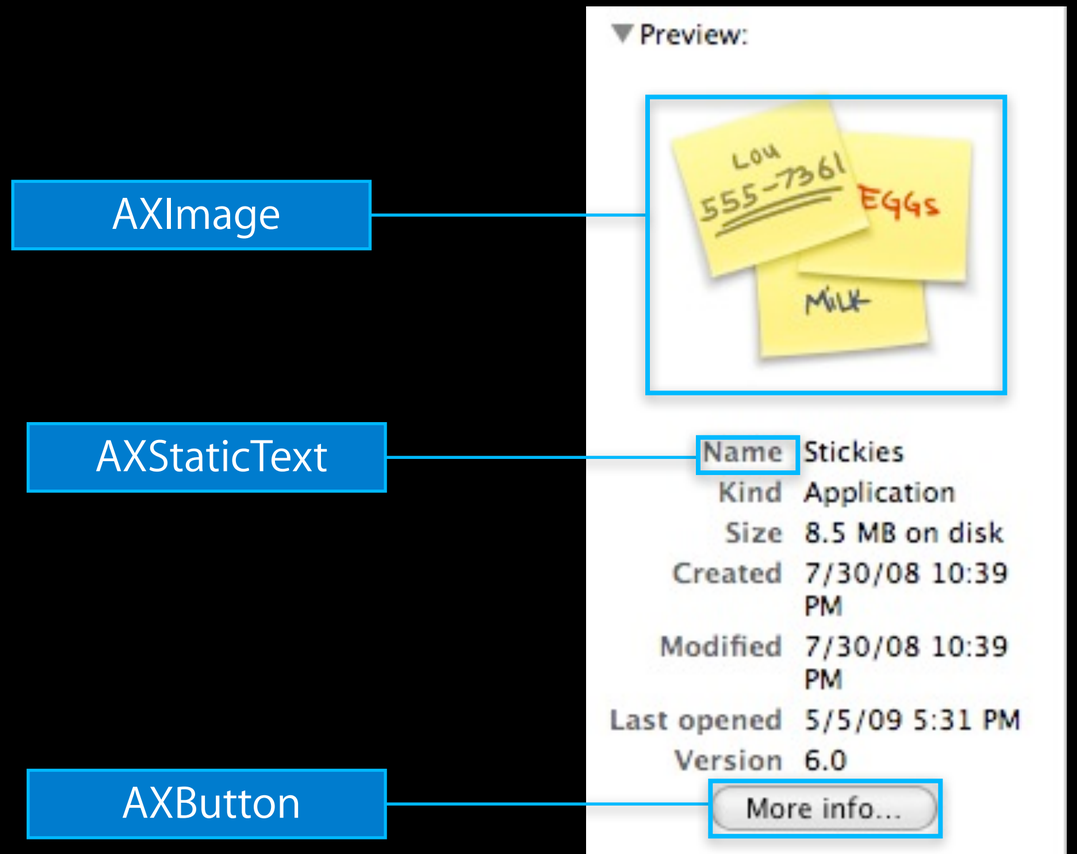
Roles and Subroles

Programmatically simple, but a big part of prior planning

- Every UI element has a role and a role description
- Some also have a subrole
- Attributes, actions, and notifications needed are based on role/subrole

Sometimes Straightforward

Obvious mapping to existing roles and subroles

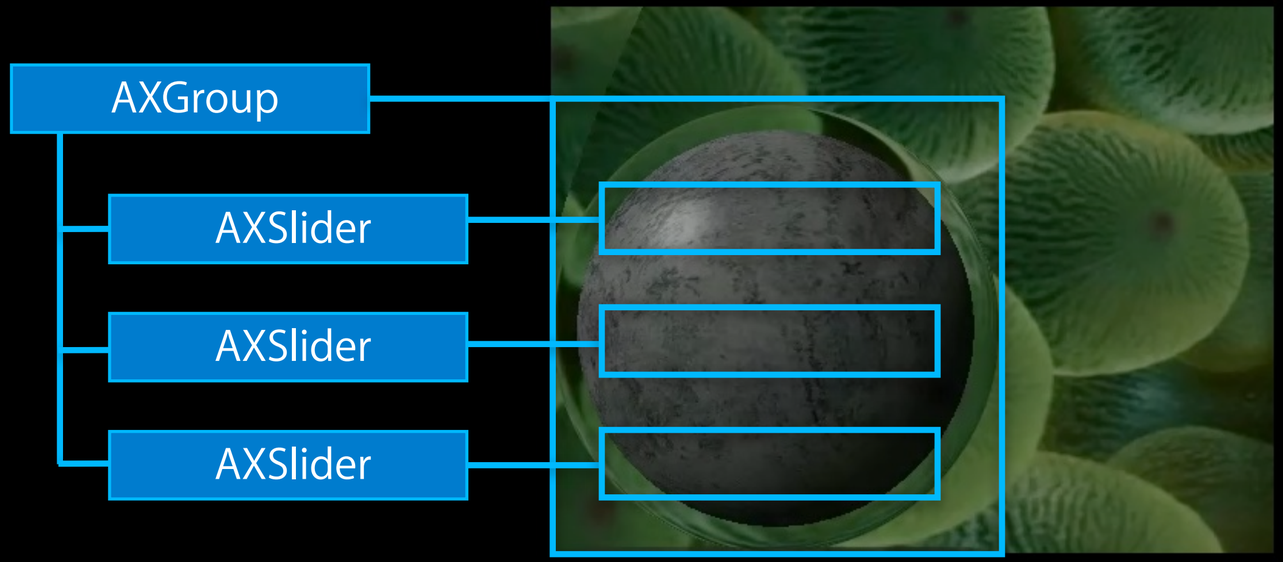


Sometimes Not

Not a clear mapping to existing roles and subroles

Sometimes Not

Solution: Map to roles that expose functionality



What to Do When There Isn't an Exact Fit?

Don't let perfect be the enemy of good

- Follow a few guidelines
  - First, try to map to visual parts as best as possible
  - If not possible, expose the functionality of the element
- Ask us and the community
  - accessibility-dev@groups.apple.com
- File enhancement request (with screen shot)
  - http://bugreporter.apple.com

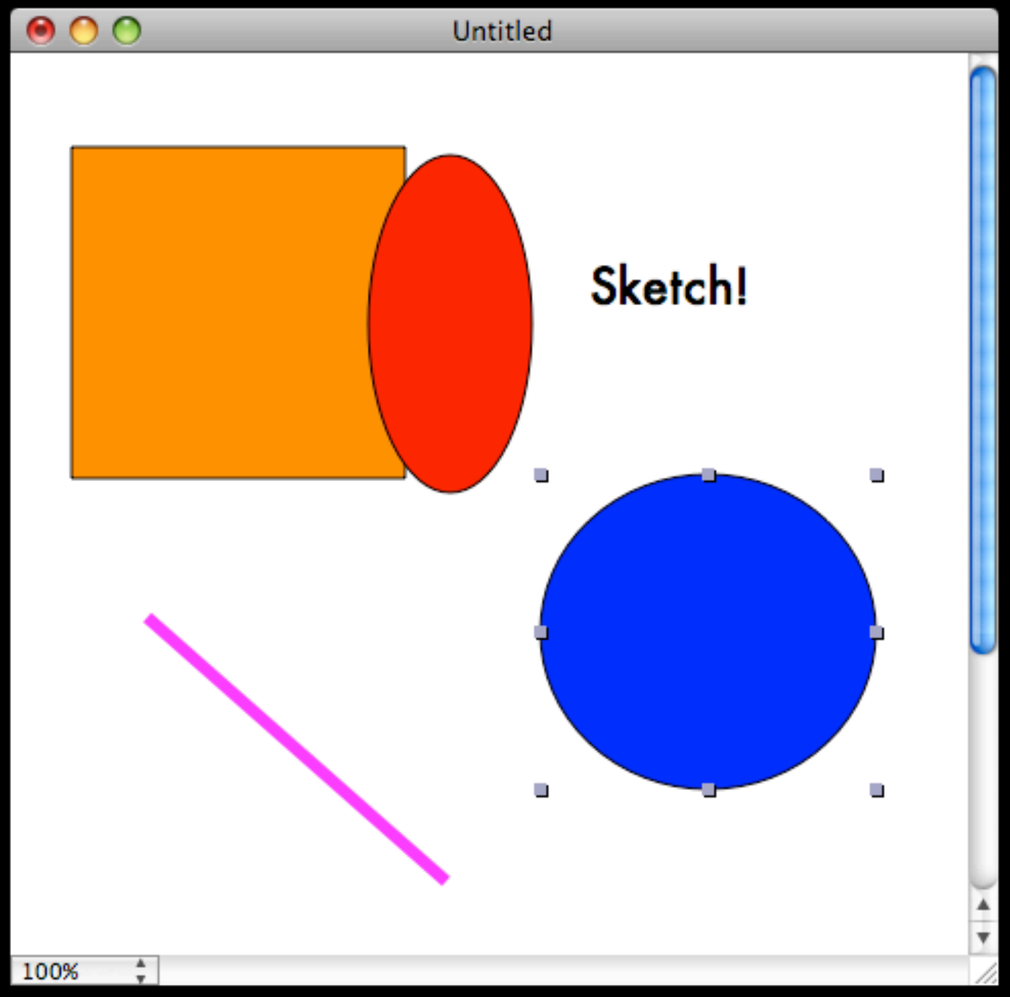
Alternate Role Descriptions

Use alternate role descriptions very sparingly

- Use only when
  - Role description does not match standard term
  - Role description will cause a lot of confusion in collaboration
  - e.g., Tabs behave like AXRadioButtons, but are typically called ‘tabs’
- Why sparingly?
  - Each new role description represents a new, different element type a user needs to learn how to interact with
  - The smaller the number of described roles, the easier the system, and your application, is to learn

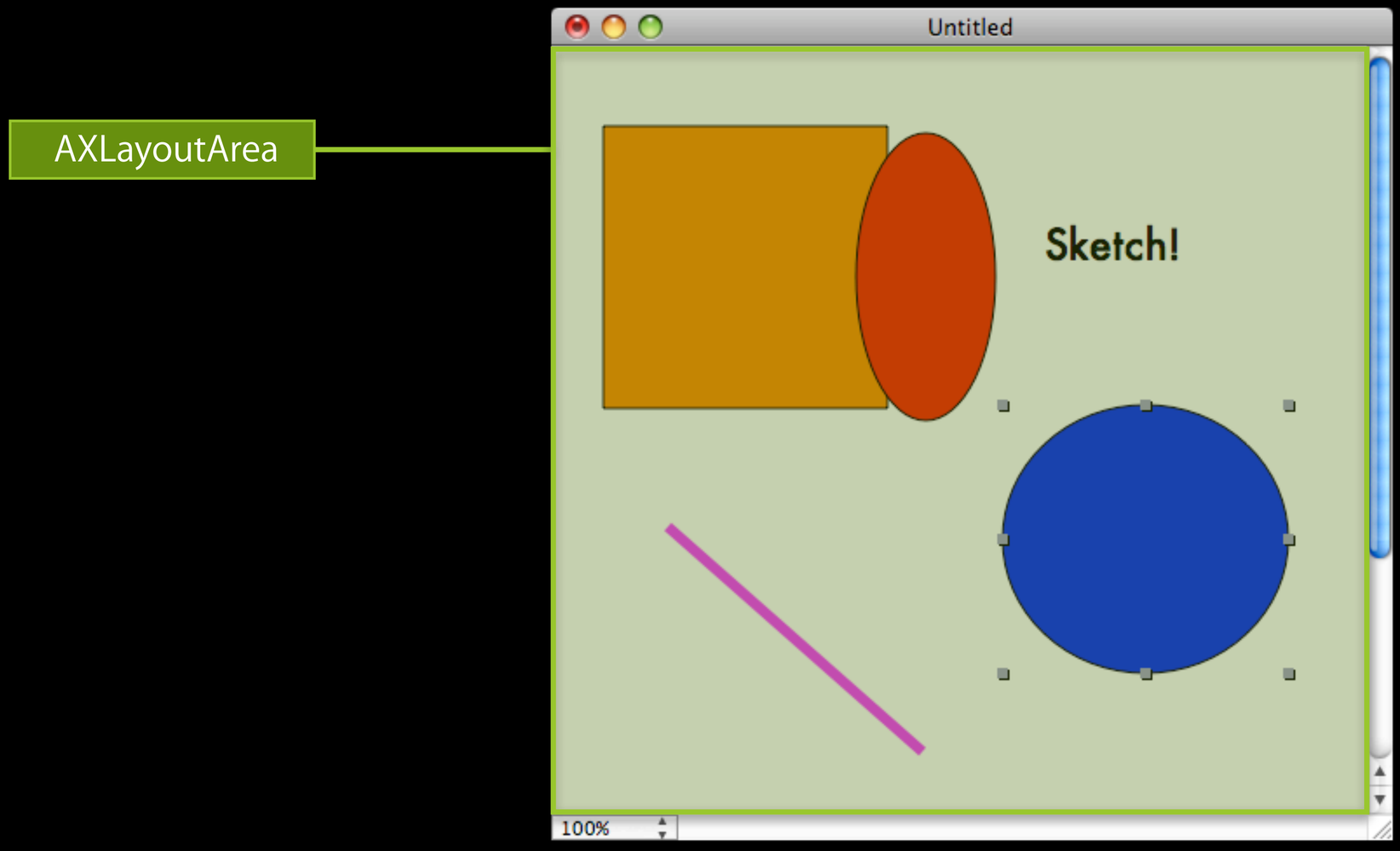
Sketch

Choosing roles and defining desired hierarchy



Sketch

Choosing roles and defining desired hierarchy

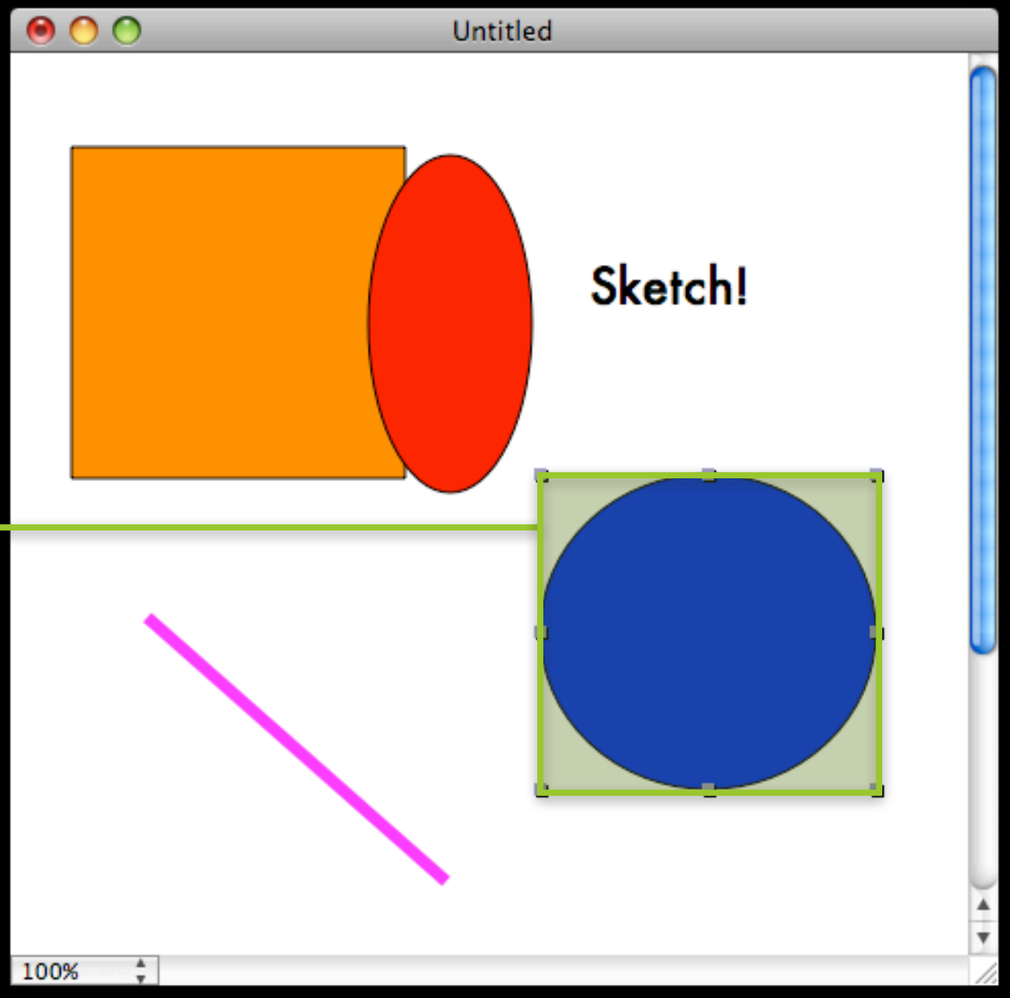


Sketch

Choosing roles and defining desired hierarchy

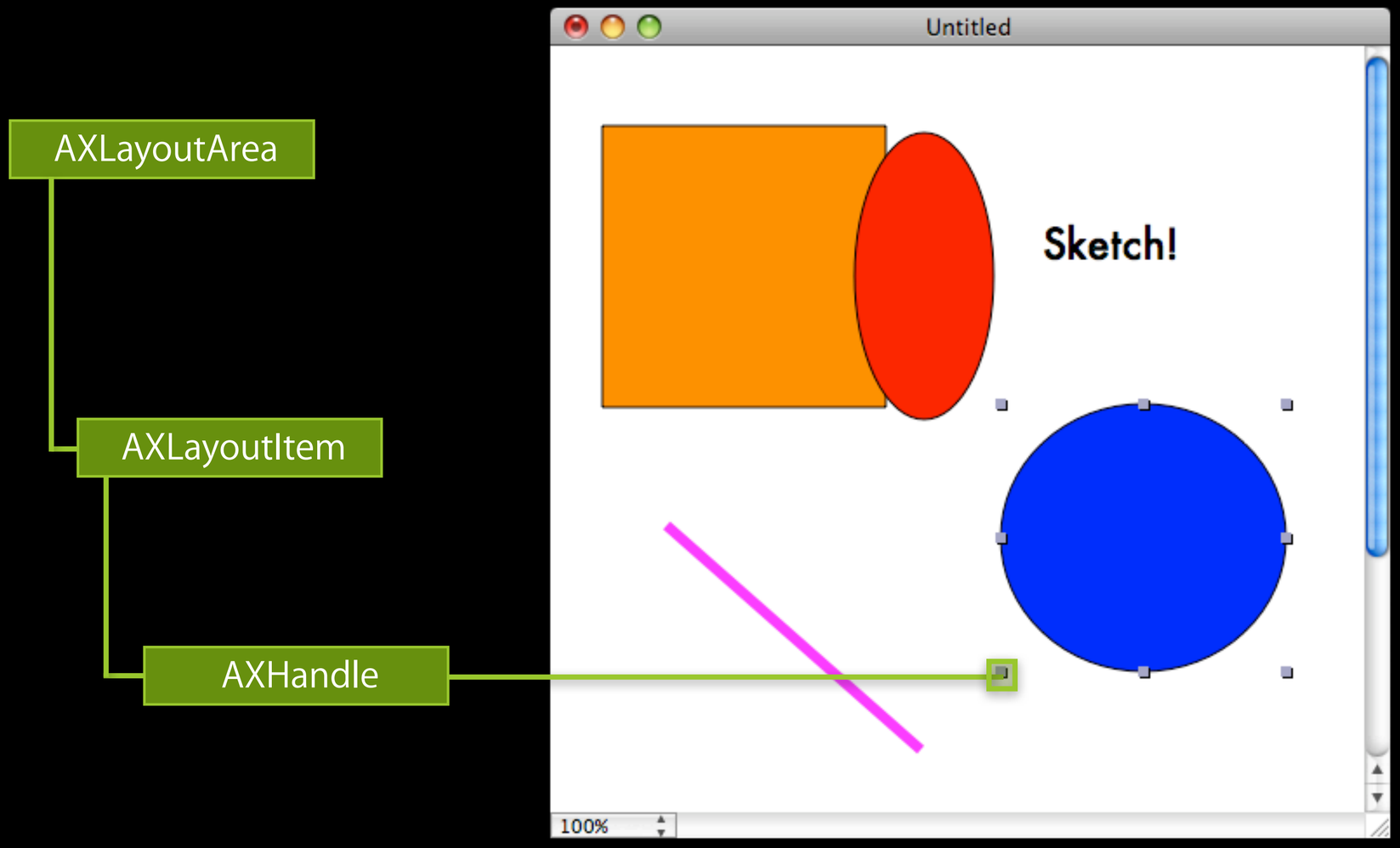
AXLayoutArea

AXLayoutItem



Sketch

Choosing roles and defining desired hierarchy



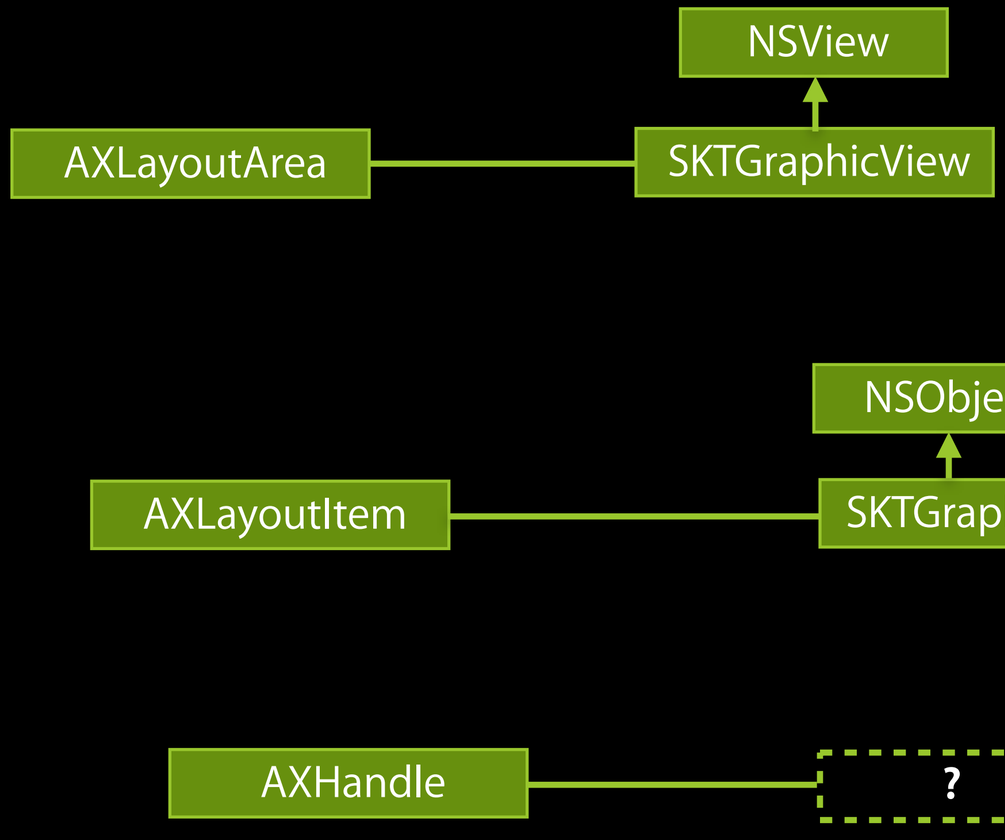
Mapping Object to Element Hierarchy

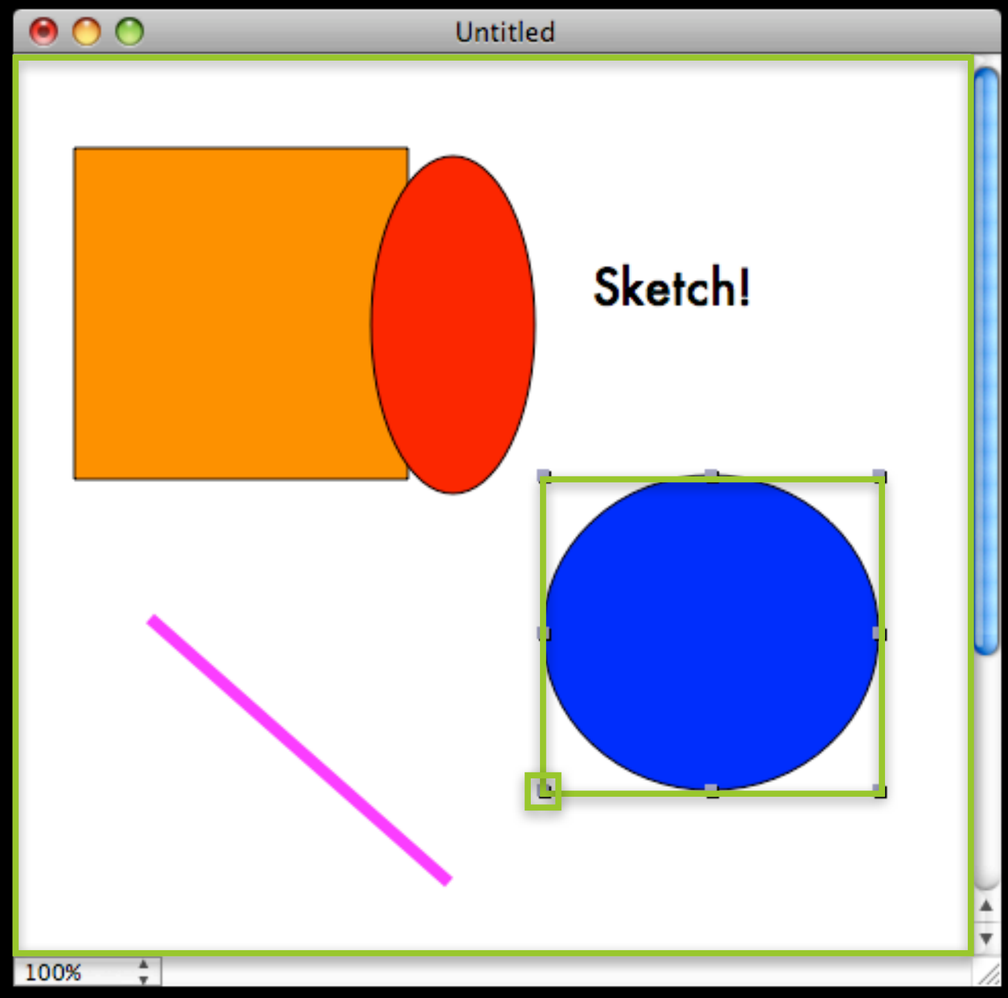
How do we get there from here?

- Review existing object hierarchy
- Determine the following
  - Which objects will be ignored
  - Which objects need to override NSAccessibility methods
  - Which objects need to implement the NSAccessibility protocol
  - Which elements don’t have an analogous object
    - Faux UI elements
    - Proxy elements

Sketch

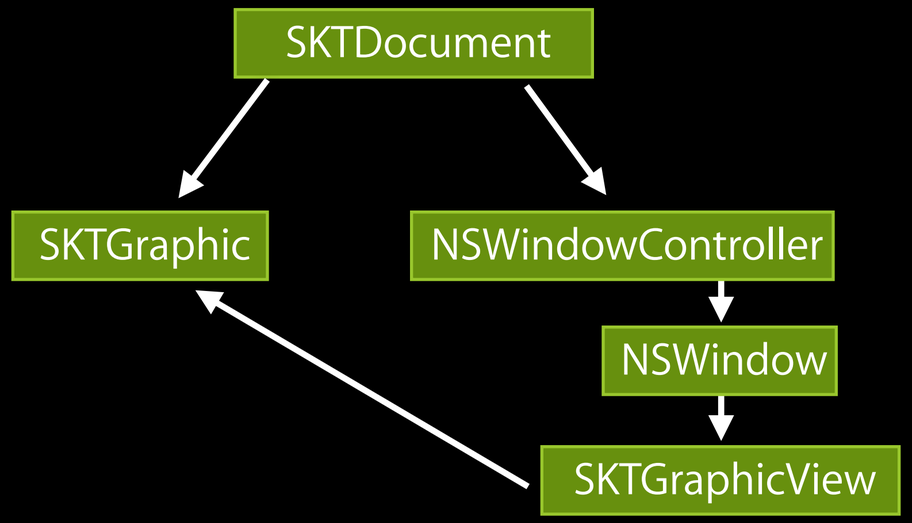
Mapping hierarchies





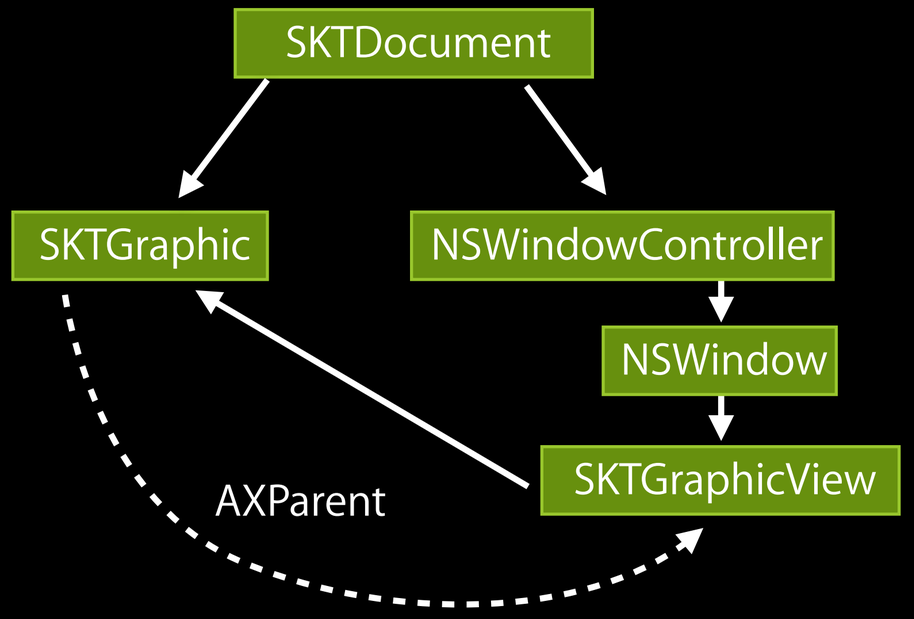
Sketch

Mapping hierarchies



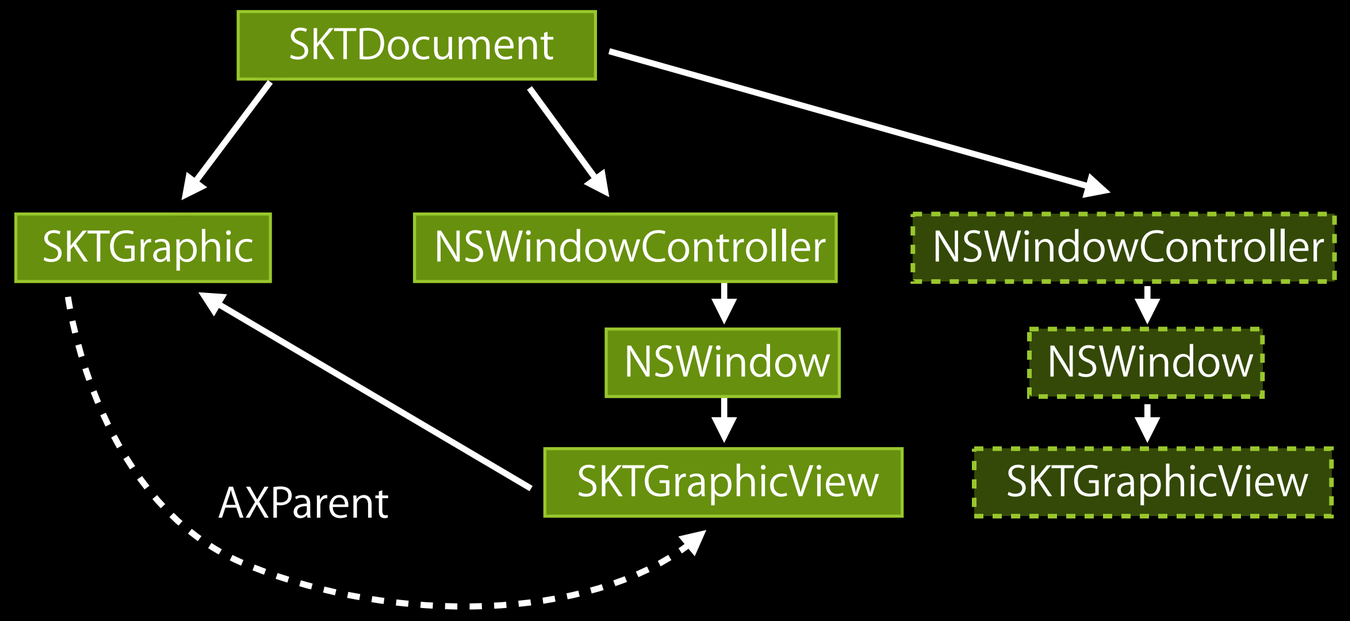
Sketch

Mapping hierarchies



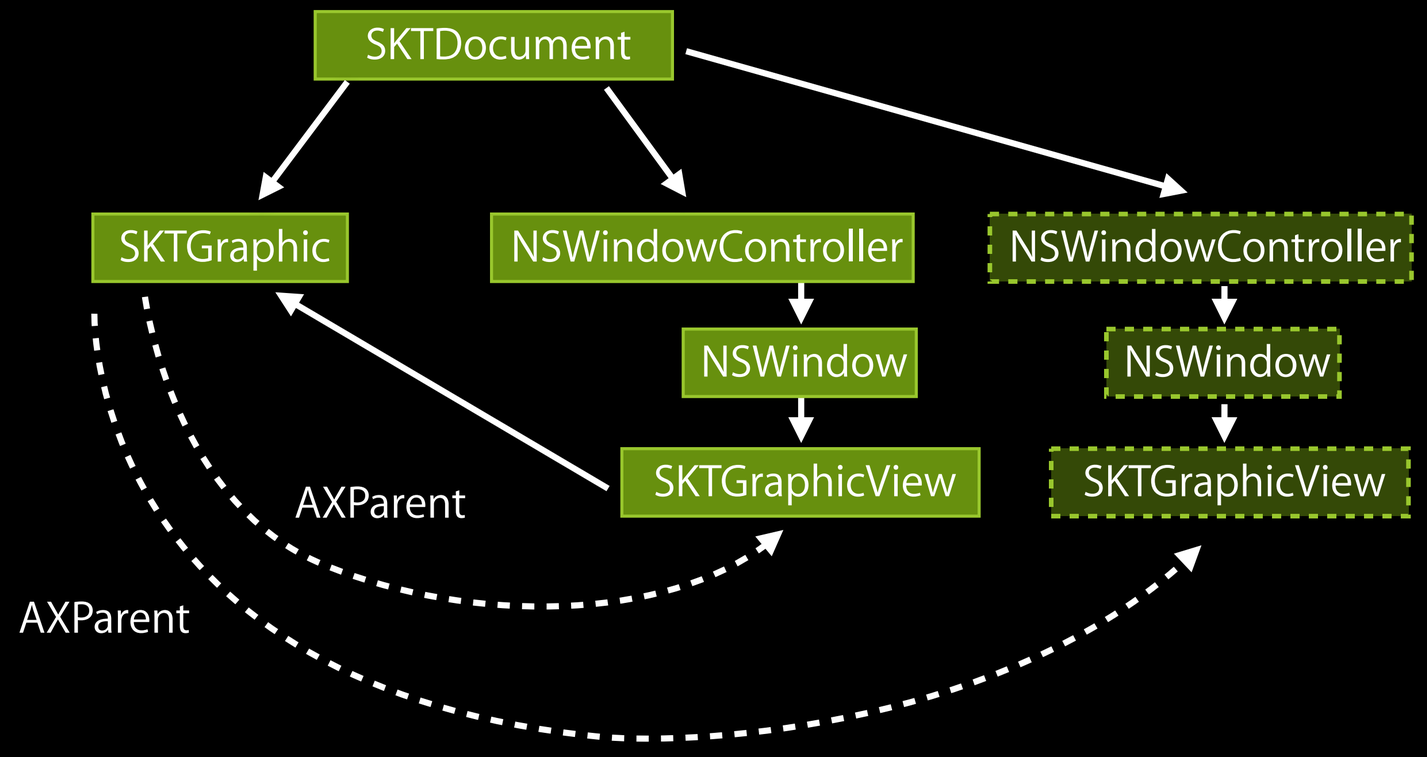
Sketch

Mapping hierarchies



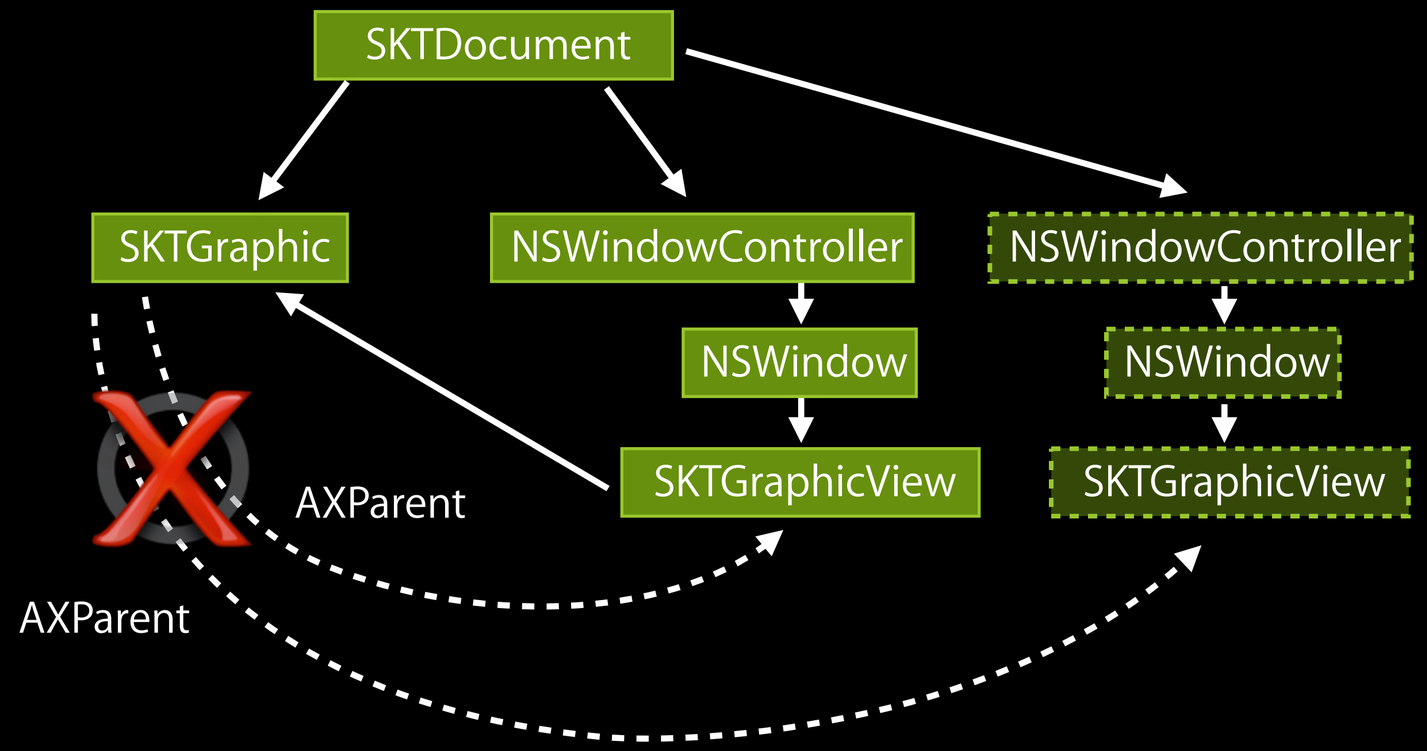
Sketch

An element can’t have two parents



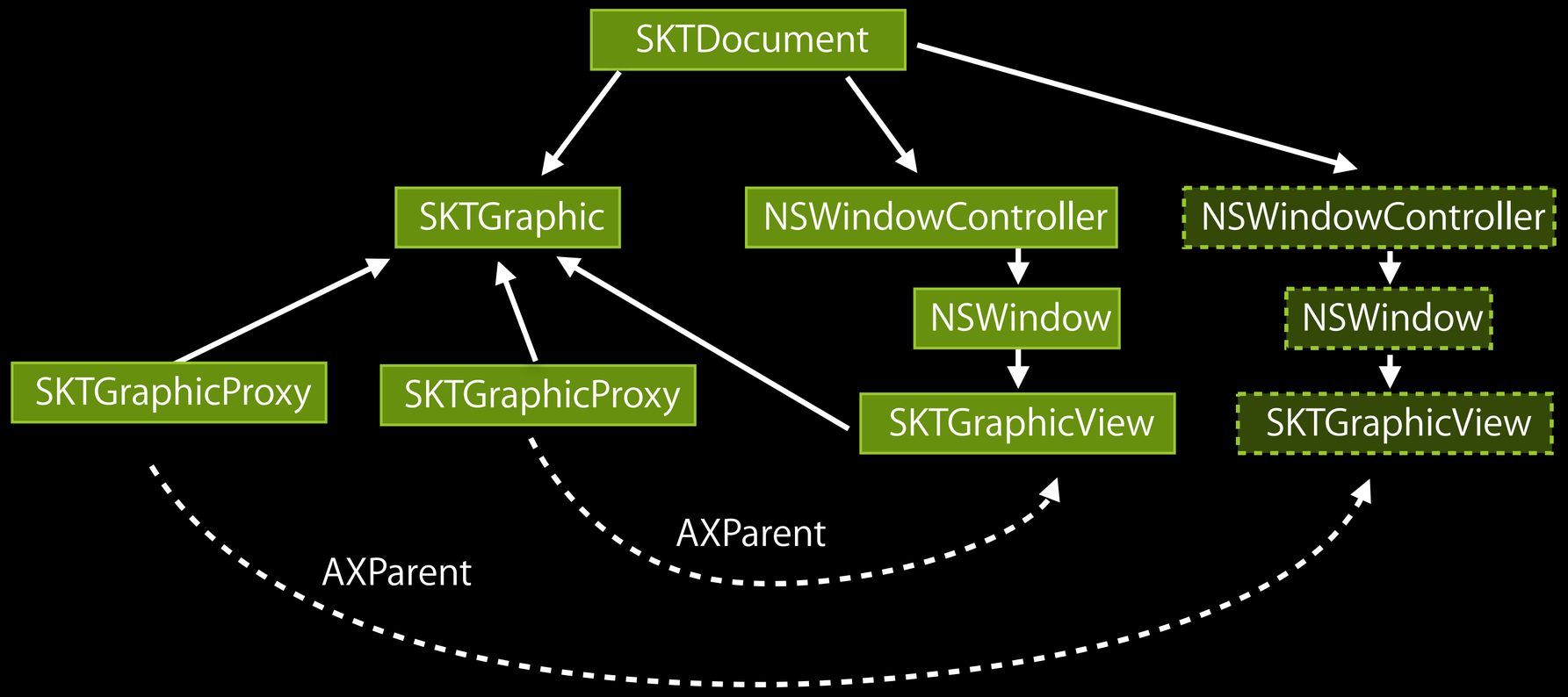
Sketch

An element can’t have two parents



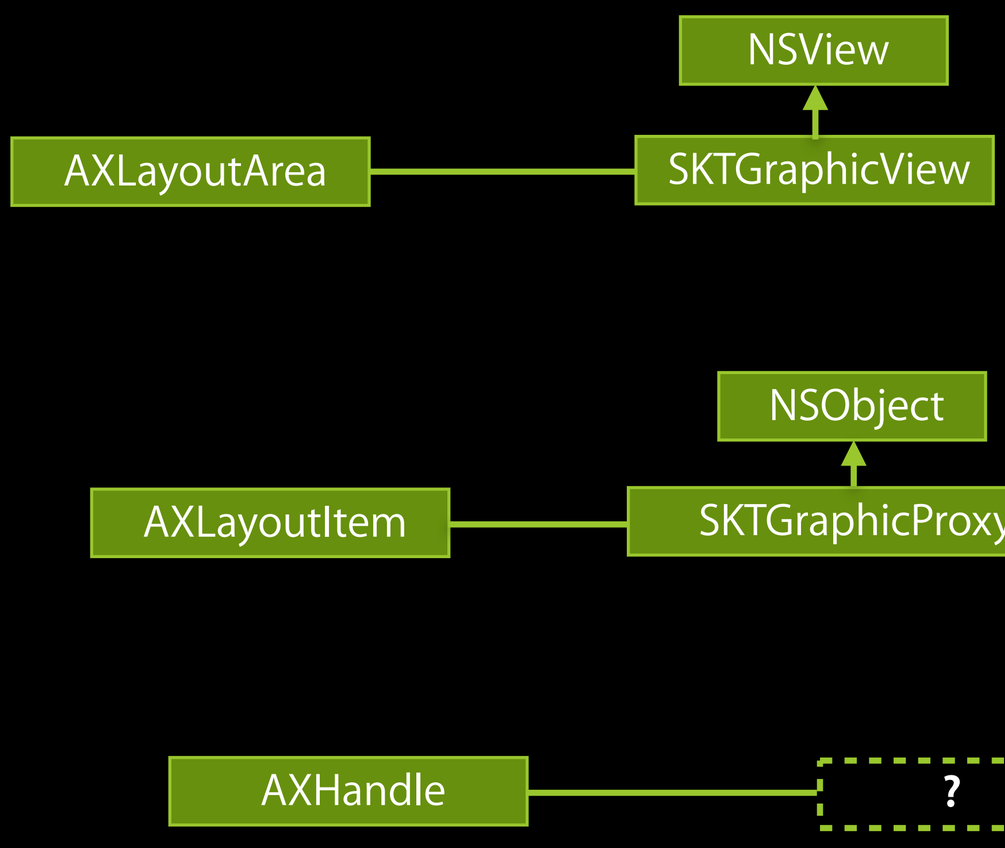
Sketch

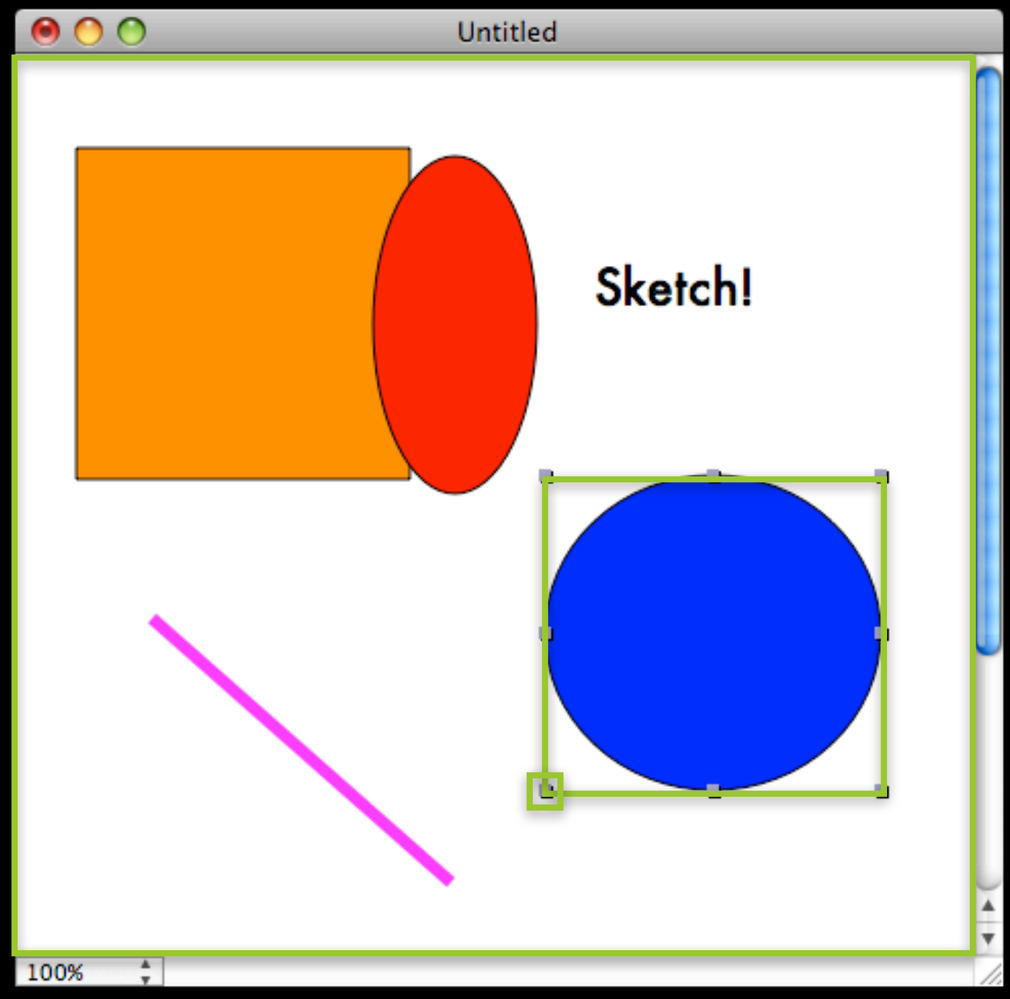
Solution: Use a proxy element object



Sketch

Mapping hierarchies





Demo

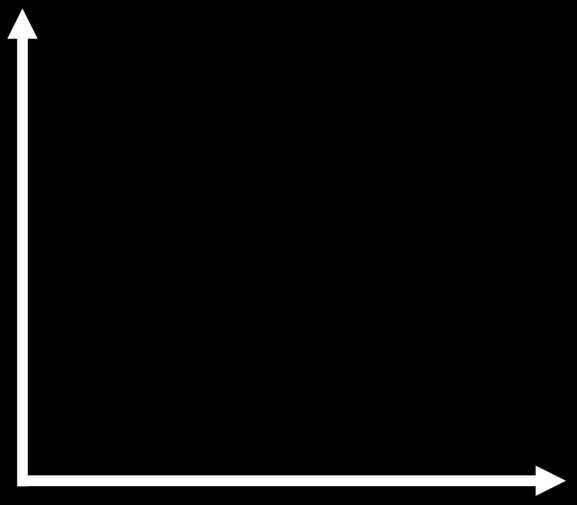
Reporting graphics in the graphic view

Things Required by Every Element

- Reference to its parent
- Direct or indirect access to its containing NSView

Accessibility Geometry

NSAccessibility uses Cocoa screen coordinates



Cocoa Screen Coordinates

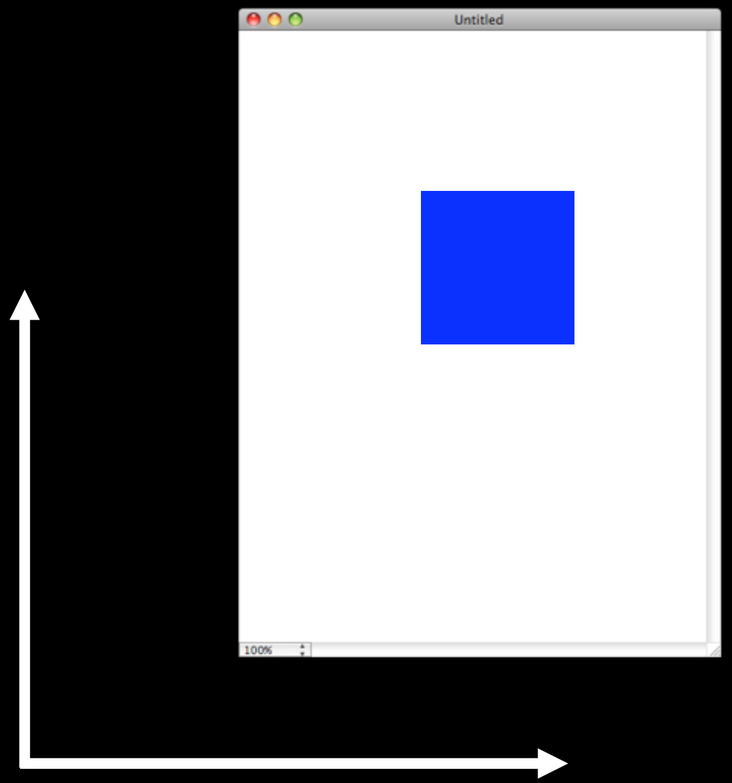
All points you receive and pass back are in Cocoa screen coordinates

AX Screen Coordinates

What you will see reported to the AX APIs, and see in Accessibility Inspector uses upper left as origin

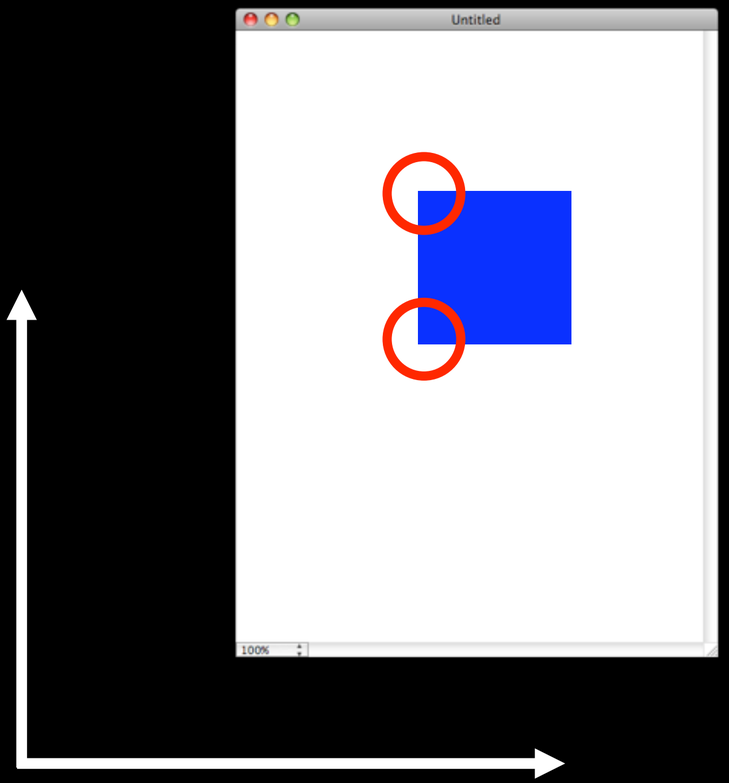
Accessibility Geometry

Translate from view to window, window to screen



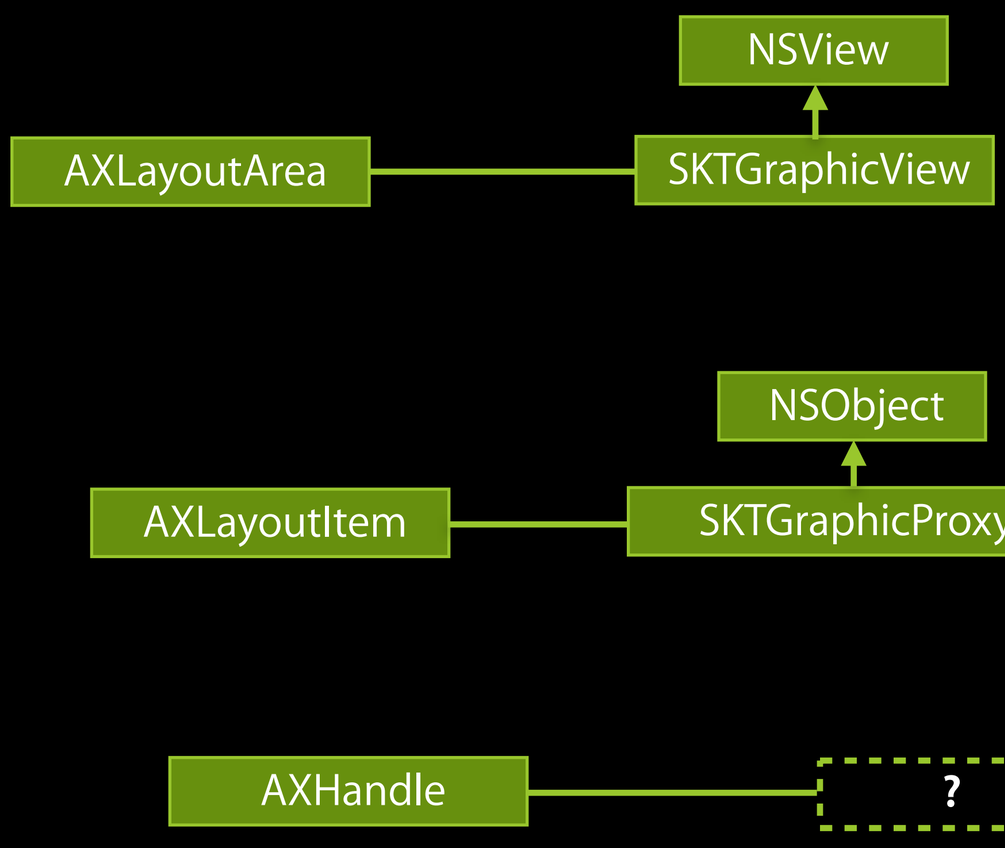
Accessibility Geometry

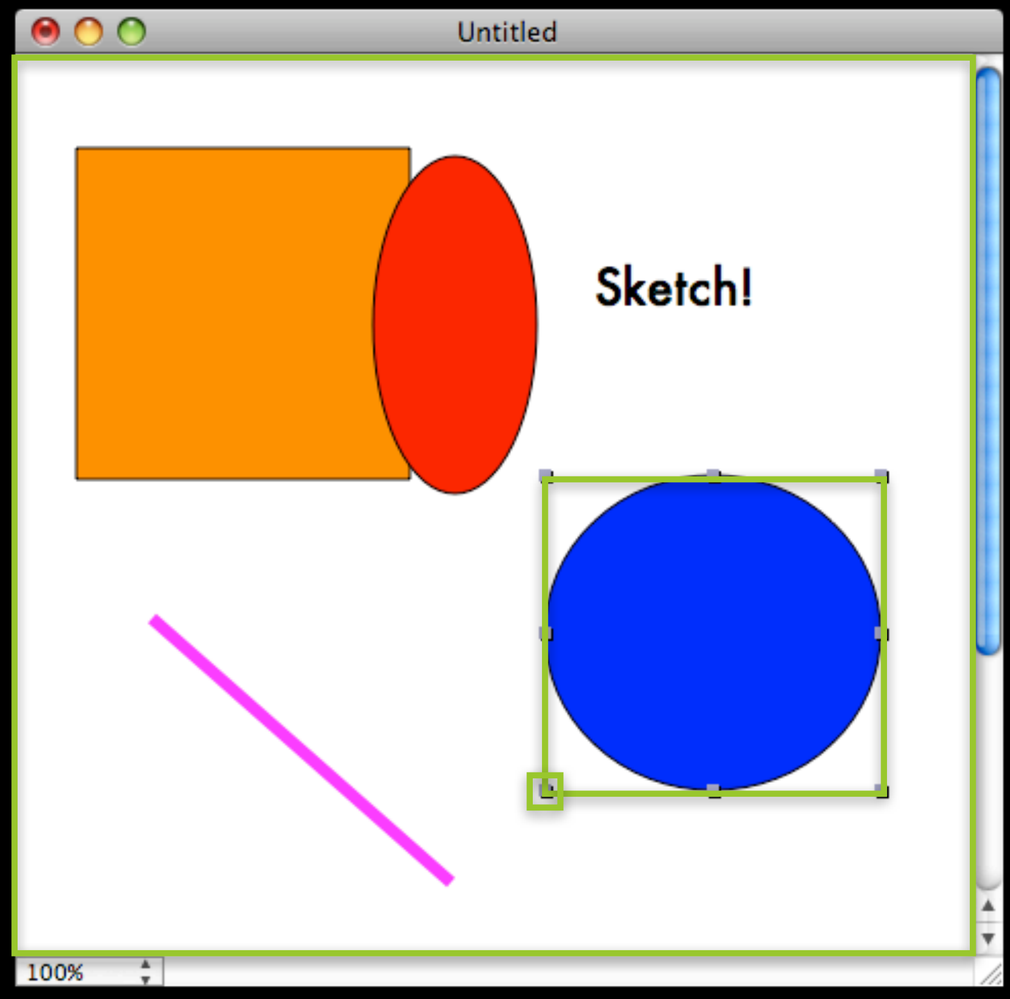
If view is flipped, you need to adjust origin point



Sketch

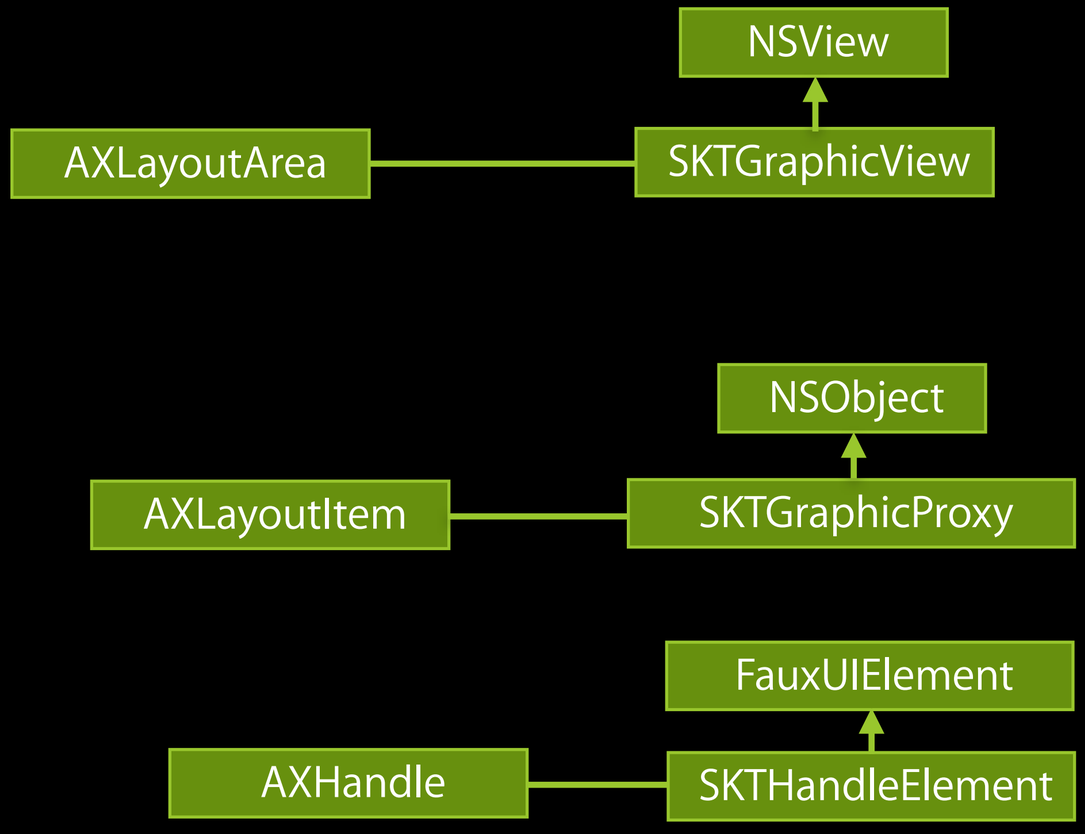
Mapping hierarchies

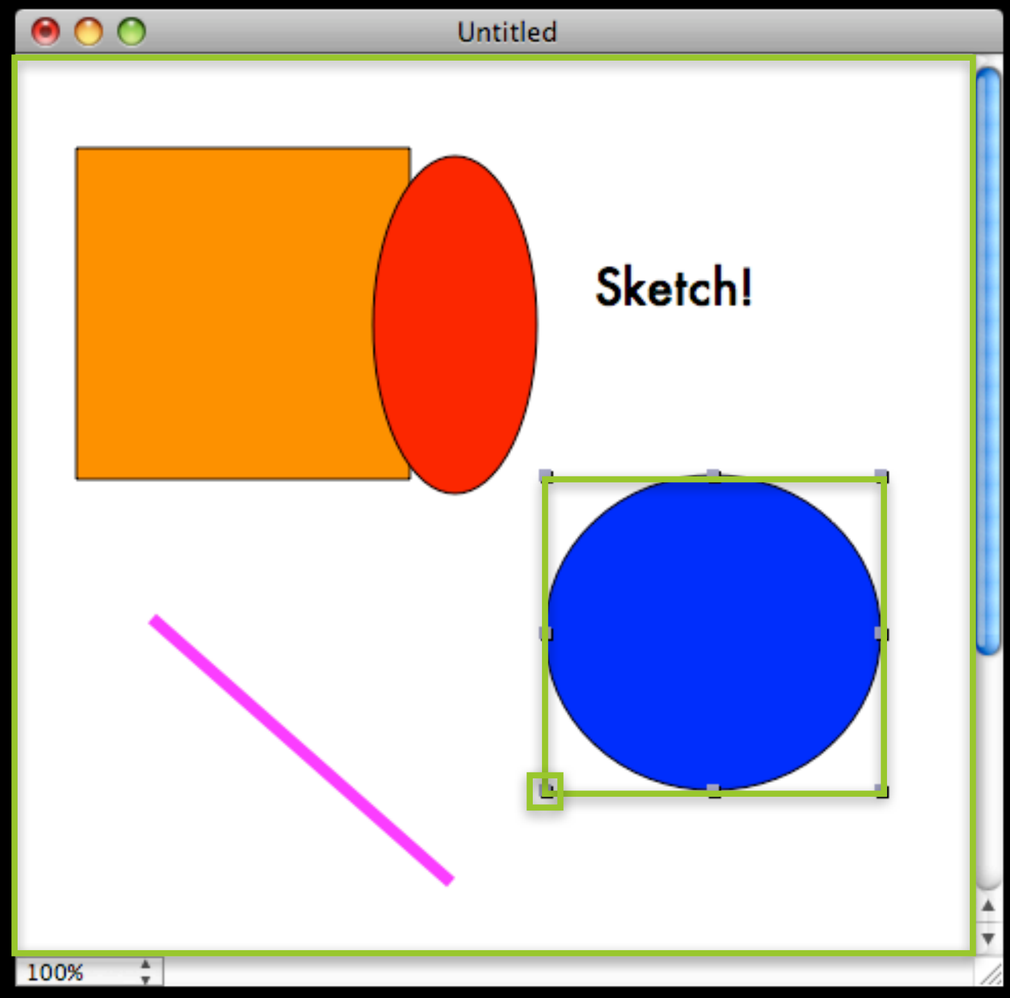




Sketch

Mapping hierarchies





FauxUIElement

- Reusable sample code class for a false UI element
- Reused as-is in Sketch from ImageMap example

Demo

Getting a handle on things

Now Comes Phase Three

Working through the remaining items

- Remaining attributes
- Focus testing
- Actions
- Test with NSAccessibilityDebugLogLevel 1

Troubleshooting

- Ensure parent reports child and vice versa
- Ensure objects report -isEqual: and -hash correctly
  - Proxy element objects
  - Faux element objects
- Use unignored element helper functions
- Set argument NSAccessibilityDebugLogLevel 1

Demo

Final Demo

More Information

Eric Hope
Application Technologies Evangelist
ehope@apple.com

Accessibility Mailing List
Public Developer List
accessibility-dev@lists.apple.com

Documentation
Mac OS X User Experience—Accessibility
http://developer.apple.com/ue/accessibility

Related Sessions

Enabling Your Mac Applications for Assistive Technologies	North Beach Wednesday 5:00PM

Labs

Mac OS X Accessibility Lab	OS X Lab D Friday 2:00PM

Q&A

WWDC Session Survey

You can rate your sessions on the WWDC Attendee Site.

Survey forms are located on each session description page at:

developer.apple.com/wwdc/attendee

WWDC09